

1 The implementation described above uses node-link data that include
2 expansion flags of links to define a tree within a graph as disclosed in copending
3 coassigned U.S. Patent Application ^{09/124,338} ~~09/EEE,EEE~~ (Attorney Docket No.
4 D/98205Q4), entitled "Node-Link Data Defining a Graph and a Tree Within the
5 Graph", with memory management as disclosed in copending coassigned U.S.
6 Patent Application ^{09/124,474} ~~09/DDD,DDD~~ (Attorney Docket No. D/98205Q3), entitled
7 "Controlling Which Part of Data Defining a Node-Link Structure is in Memory",
8 both incorporated herein by reference, but the invention could be implemented
9 with a node-link structure defined in any other appropriate way, and loaded into
10 memory in any appropriate way.

11 The implementation described above employs a directed graph data
12 structure in which a link is represented as an item in two linked lists, one for the
13 outgoing links from its from-node and one for the incoming links to its to-node.
14 Any other suitable data structure could be employed.

15 The implementation described above can handle directed graphs,
16 including cyclic directed graphs, but the invention could be implemented for
17 other types of graphs by converting other types of links to appropriate
18 combinations of directed links or by otherwise providing a protocol for mapping
19 the structure of a graph to a tree. For example, an undirected link between two
20 nodes could be converted to a pair of directed links between the same nodes or
21 could be assigned a direction based on an appropriate criterion. In general, a
22 representation in which all undirected links have been converted to a pair of

1 by obtaining a layout with the least overlap in areas. Layout results could be
2 cached to minimize repetitive computation.

3 In the implementation described above, a node-link structure laid out in
4 the hyperbolic plane is then mapped into the unit disk and then painted in
5 accordance with copending coassigned U.S. Patent Application ^{09/124,529} ~~09/666,666~~
6 (Attorney Docket No. D/98205Q2), entitled "Mapping a Node-Link Structure to a
7 Rendering Space Beginning from any Node", incorporated herein by reference,
8 but a node-link structure laid out in accordance with the invention could be laid
9 out in any other appropriate negatively curved space, and then be handled in
10 any other appropriate way, with or without mapping, or mapped and presented in
11 any other appropriate way, including mapping it into any other appropriate
12 rendering space and presenting it in any other appropriate display space,
13 including three-dimensional rendering and display spaces.

14 The implementation described above determines whether to lay out a
15 node's children by comparing an angle displacement obtained for the node with
16 a previous angle displacement, but the invention could be implemented by laying
17 out all descendants of each laid out node or by applying any other suitable
18 criterion to determine which elements to lay out.

19 The implementation described above is suitable for laying out elements of
20 a tree. The invention could be used to lay out elements of other types of node-
21 link structures, such as graphs in general.

Change(s) applied
to document,
/J.L.B./
1/6/2012

09/124,529

1 node-link structure like the representations disclosed in Lamping et al., US-A-
2 5,619,632 and in copending coassigned U.S. Patent Application ~~09/AAA,AAA~~
3 (Attorney Docket No. D/98205), entitled "Presenting Node-Link Structures with
4 Modification", both incorporated herein by reference. The invention could,
5 however, be implemented with or without navigation signals; for example,
6 elements could be moved around in response to different sortings of the children
7 of a node or in response to the application of different filters to elements of a
8 structure. Also, the invention could be implemented with any appropriate type of
9 expand and contract signals or other navigation signals, including signals
10 resulting from external queries, selection of a menu entry-like item requesting
11 expansion below an indicated node or link, or selection of a menu entry-like item
12 requesting expansion below the current focus. The navigation signals could
13 instead relate to an illusory space like those produced by videogames or virtual
14 reality environments or a presentation space other than a display and navigation
15 signals could instead be produced by any appropriate user input device,
16 including other kinds of pointing devices and other kinds of devices for receiving
17 alphanumeric or linguistic input such as voice, gestures, or other modes of user
18 input. Further, the invention could be implemented with other types of
19 representations of node-link structures. The invention could be implemented
20 without animation or with any appropriate animation techniques.

21 The implementation described above obtains nearby relationship data
22 relating to relationships among a parent, its children, and its grandchildren, but

1 root node at the new orientation before mapping and painting, in box 404. The
2 root node can be laid out as described above in relation to box 352 in Fig. 6, but
3 with the new orientation. The new orientation will then be used in mapping,
4 changing the orientation of the representation.

5 If the change is a non-animated edit, as could occur in response to a
6 stretch event, a drag event, a bookmark event, or a click event if edits are
7 pending, walker routines 222 first set up a list of remove edits, in box 410, and
8 then lay out the remove edits before mapping and painting, in box 412. Then,
9 walker routines 222 set up a list of add edits, in box 414, and then lay out the
10 add edits before mapping and painting, in box 416.

11 In the current implementation, the lists of edits are set up based on edit
12 source lists that are maintained by various routines in memory 214, including
13 grapher routines 220 and painter routines 224. Also, the current implementation
14 relates to a tree defined by expanded links, as explained in copending
15 coassigned U.S. Patent Application ^{09/124,338} ~~09/EEE,EEE~~ (Attorney Docket No.
16 D/98205Q4), entitled "Node-Link Data Defining a Graph and a Tree Within the
17 Graph", incorporated herein by reference. One pair of edit source lists,
18 designated "CollapsedLinks" and "ExpandedLinks" herein, includes edits for
19 links selected by contract requests and expand requests, respectively, and can
20 therefore be set up in box 330 in Fig 5. The other pair, designated
21 "RemovedLinks" and "AddedLinks" herein, includes edits for links that are

to document,

09/124,338

/J.L.B./

1/6/2012

1 ~~09/EEE,EEE~~ (Attorney Docket No. D/98205Q4), entitled "Node-Link Data
2 Defining a Graph and a Tree Within the Graph", incorporated herein by
3 reference.

4 The test in box 380 applies an appropriate criterion to determine whether
5 to continue layout to the next generation of nodes. As noted above in relation to
6 box 374, the criterion can be whether any child node's angle has been modified
7 by more than a small angular difference, such as 0.00001. If so, layout should
8 continue.

9 In box 382, walker routines 222 push the ID of each child node that is
10 expanded or not a leaf onto the front of the queue. Other child nodes could be
11 marked walked in box 382, since they do not have children that will be laid out.
12 When box 382 is completed or if the test in box 380 determines not to continue
13 or the test in box 370 determines that the node is already walked, the back node
14 on the queue is popped, in box 384, before returning to box 360.

15 Fig. 7 illustrates how layout of a changed node-link structure can be
16 performed in boxes 312, 324, and 334 in Fig. 5. In each case, layout begins in
17 response to a call that leads to layout and mapping, as shown in box 400. As
18 illustrated by the branch in box 402, however, the manner in which layout is
19 performed depends on the type of change being made in the node-link structure.

20 If the change is a change in orientation of the root node in response to an
21 orientation event, walker routines 222 can call math routines 226 to lay out the

1 U. S. Patent 5,590,250, incorporated herein by reference. This function, referred
2 to herein as "RoomAvailable", starts with a distance D that has been moved into
3 a wedge and an angle Φ that is half of the wedge. RoomAvailable returns a
4 distance to the edge of the wedge that is calculated by first obtaining the ratio
5 $(1-D^2)/2D$, and by then dividing the ratio by $\sin\Phi$ to obtain an initial distance S.
6 RoomsAvailable then returns the distance $((S^2-1)^{1/2}-S)$. To obtain a child's area,
7 RoomAvailable is called with the same distance and angle that were used in
8 calling InsideAngle, as described above. The distance returned by
9 RoomAvailable is saved as a measure of the child's area.

10 Although the software implementation described above can save
11 additional data, it is based on the discovery that only two items of data need to
12 be stored for each node in order to be able to perform layout and mapping as
13 described herein and in relation to Fig. 6 of copending coassigned U.S. Patent
14 Application ^{09/124,529} ~~09/000,000~~ (Attorney Docket No. D/98205Q2), entitled "Mapping a
15 Node-Link Structure to a Rendering Space Beginning from any Node",
16 incorporated herein by reference. One item indicates a distance or position
17 displacement from the node to its children nodes in the hyperbolic plane. The
18 other is an angle displacement in the hyperbolic plane between the extension of
19 the incoming link to the node's parent and outgoing link from the parent to the
20 node. These two items of data, or a handle that can be used to access them,
21 can be included in a link's data item in a directed graph data structure as
22 illustrated in Fig. 6 of copending coassigned U.S. Patent Application

Change(s) applied
to document,
/J.L.B./
1/6/2012

1 deletions and insertions. The technique has been successfully implemented to
2 produce object constancy during these movements.

3 After a representation is provided in box 316 or after an animation
4 sequence is completed in box 322 or 332, another event can be received in box
5 302, as indicated by the circles labeled “A” in Fig. 5.

6 Animation details relating to the loops that begin in boxes 322 and 332
7 are discussed in more detail in copending coassigned U.S. Patent Application
8 ~~09/124,528~~ ~~09/AAA,AAA~~ (Attorney Docket No. D/98205), entitled "Presenting Node-Link
9 Structures with Modification", incorporated herein by reference.

10 C.3. Layout

Fig. 6 shows how layout can be initially performed in box 300 in Fig. 5.

Fig. 7 shows how layout of a changed node-link structure can be performed in boxes 312, 324, and 334.

14 As shown in box 350, walker routines 222 begin initial layout by obtaining
15 the root node ID and using it to access data relating to the root node in directed
16 graph data structure 232. In box 352, walker routines 222 lay out the root node
17 by making a call to math routines 226 with an angle width. This could be any
18 suitable angle that produces a desirable result. The angles 2π and $\pi/2$ have
19 been successfully used, with 2π appropriate for a center layout style and with
20 $\pi/2$ appropriate for a top, bottom, right, or left layout style. An interface could
21 also be provided to modify this angle to obtain desirable results.

1 a swap of display buffers can be performed to present the tree as painted, thus
2 providing a representation of the graph.

3 When a new edit is marked in box 328 by painter routines 224, the new
4 edit is laid out during the next iteration, in box 324. As a result, the animated
5 sequence of representations, rather than showing a static node-link structure as
6 in U.S. Patent 5,629,632, shows a dynamic node-link structure. The edits,
7 however, serve primarily to add features representing new nodes along the outer
8 perimeter of the representation as the representation makes the transition from
9 the previous position to the destination position. As a result, the added features
10 do not interfere with or reduce the perception of object constancy for features
11 representing other elements.

12 An insert/delete event could result when the user requests expansion or
13 contraction of a node or requests some other modification of the graph or the
14 tree. An insert/delete event could also be received in the form of a call, and
15 could thus provide a mechanism for automatic modification of the graph or tree
16 without concurrent human control.

17 In response to an event of this type, client 212 can first make appropriate
18 calls to routines in memory 214 to determine whether the requested modification
19 of the graph or tree is acceptable, in box 330. For example, a technique for
20 determining whether an expand signal is acceptable is described in relation to

21 Fig. 7 of copending coassigned U.S. Patent Application 09/124,338 (Attorney

09/124,338

~~09/EEE,EEE~~

Change(s) applied
to document,

/J.L.B./

1/6/2012

1 In box 312, walker routines 222 could first perform any necessary layouts
2 in the hyperbolic plane, and could also lay out any pending edits of the tree.
3 Then, in box 314, walker routines 222 could map the tree into the unit disk,
4 beginning with a starting node at a starting position, in the manner described in
5 copending coassigned U.S. Patent Application ^{09/124,529} ~~09/000,000~~ (Attorney Docket
6 No. D/98205Q2), entitled "Mapping a Node-Link Structure to a Rendering Space
7 Beginning from any Node", incorporated herein by reference. For example, in
8 response to a dragging event, the starting node could be the nearest node
9 identified in box 310 and the starting position could be the next position along
10 the path of motion. The starting node and starting position previously used for
11 mapping could be used in response to an orientation or stretch event.

12 When the tree has been mapped, painter routines 224 can be called to
13 paint the mapped tree in a display buffer, in box 316. During painting, painter
14 routines 224 can mark new edits that occur in the tree as a result of node
15 creation as described in copending coassigned U.S. Patent Application
16 ^{09/124,338} ~~09/000,000~~ (Attorney Docket No. D/98205Q4), entitled "Node-Link Data
17 Defining a Graph and a Tree Within the Graph", incorporated herein by
18 reference. Each edit can be marked by setting a flag or storing other
19 appropriate data. When painting is completed, a swap of display buffers can be
20 performed to present the tree as painted, thus providing a representation of the
21 graph.

1 Node position data 232, which can be linked to or included within directed
2 graph data structure 230, can include positions of nodes in a negatively curved
3 space such as a hyperbolic plane and in a rendering space such as a two-
4 dimensional unit disk. Node position data 232 can be accessed by routines in
5 program memory 214.

6 The routines in program memory 214 can also access various
7 miscellaneous data structures 234. Data structures 234 may, for example,
8 include an extra data structure for mapping from a pair of node IDs to a link ID,
9 implemented as a standard heap; this extra data structure allows lookup and
10 insertion of a link ID in constant expected time.

11 C.2. Responding to Events

12 Fig. 5 shows how the system of Fig. 4 can respond to events by
13 presenting representations of a graph.

14 In box 300, client 212 begins by obtaining a starting graph and by loading
15 an initial set of elements into memory, such as through calls to create nodes as
16 described in copending coassigned U.S. Patent Application ^{09/124,474} ~~09/124,474~~
17 (Attorney Docket No. D/98205Q3), entitled "Controlling Which Part of Data
18 Defining a Node-Link Structure is in Memory", incorporated herein by reference.
19 Expansion flags define a tree within the initial set of elements, as described in
20 copending coassigned U.S. Patent Application ^{09/124,338} ~~09/124,338~~ (Attorney Docket
21 No. D/98205Q4), entitled "Node-Link Data Defining a Graph and a Tree Within

1 information about a directed graph, which could be a combination of routines
2 and data stored in memory 210 or could be independent of memory 210 as
3 shown. For example, processor 202 could communicate with client 212 through
4 a network.

5 The routines stored in program memory 214 can be grouped into several
6 functions. Grapher routines 220 create and modify a data structure representing
7 the directed graph defined by the information from client 212. Walker routines
8 222 respond to navigation signals and other user signals from keyboard 206 and
9 mouse 208 by obtaining information from the directed graph data structure.
10 Painter routines 224 provide signals to display 204 to cause it to present
11 representations of the directed graph data structure. Math routines 226 can be
12 called to obtain positions of elements of the directed graph in a layout space.

13 Data memory 216 in turn contains data structures accessed by processor
14 202 during execution of routines in program memory 214. Directed graph data
15 structure 230, as noted above, can be created and modified by grapher routines
16 220 and can also be accessed by walker routines 222 and painter routines 224.

Further details about the implementation of directed graph data structure
are set forth in copending coassigned U.S. Patent Applications
09/124,474
~~09/000,000~~ (Attorney Docket No. D/98205Q3), entitled "Controlling Which Part
of Data Defining a Node-Link Structure is in Memory", and ~~09/000,000~~
(Attorney Docket No. D/98205Q4), entitled "Node-Link Data Defining a Graph
and a Tree Within the Graph", both incorporated herein by reference.

18